

AFRL-IF-RS-TR-2001-204
Final Technical Report
September 2001



INFORMATION ASSURANCE TECHNOLOGIES FOR THE GLOBAL COMMAND AND CONTROL SYSTEM (GCCS) LEADING EDGE SERVICES (LES)

Secure Computing Corporation

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. F507/J318

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

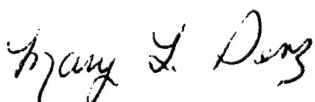
Copyright © 2000, Secure Computing Corporation. All Rights Reserved. This material may be reproduced by or for the U.S. Government pursuant to the copyright license under the clause at DFARS 252.227-7013 (Oct.88).

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

20020610 032

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2001-204 has been reviewed and is approved for publication.

APPROVED: 

MARY L. DENZ
Project Engineer

FOR THE DIRECTOR:



WARREN H. DEBANY, Technical Advisor
Information Grid Division
Information Directorate

If your address has changed or if you wish to be removed from the Air Force Research Laboratory Rome Research Site mailing list, or if the addressee is no longer employed by your organization, please notify AFRL/IFGB, 525 Brooks Road, Rome, NY 13441-4505. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

INFORMATION ASSURANCE TECHNOLOGIES FOR THE GLOBAL
COMMAND AND CONTROL (GCCS) LEADING EDGE
SERVICES (LES)

Richard O'Brien

Contractor: Secure Computing Corporation
Contract Number: F30602-97-C-0245
Effective Date of Contract: 22 June 1997
Contract Expiration Date: 31 October 2000
Short Title of Work: Information Assurance for the
GCCS LES
Period of Work Covered: Jun 97 - Oct 00

Principal Investigator: Richard O'Brien
Phone: (651) 628-2765
AFRL Project Engineer: Mary L. Denz
Phone: (315) 330-2030

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION
UNLIMITED.

This research was supported by the Defense Advanced Research
Projects Agency of the Department of Defense and was monitored
by Mary Denz, AFRL/IFGB, 525 Brooks Road, Rome, NY.

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202 4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE OCTOBER 2001		3. REPORT TYPE AND DATES COVERED Final Jun 97 - Oct 00
4. TITLE AND SUBTITLE INFORMATION ASSURANCE TECHNOLOGIES FOR THE GLOBAL COMMAND AND CONTROL (GCCS) LEADING EDGE SERVICES (LES)			5. FUNDING NUMBERS C - F30602-97-C-0245 PE - 63760E PR - F025 TA - 11 WU - 32	
6. AUTHOR(S) Richard O'Brien				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Secure Computing Corporation 2675 Long Lake Road Roseville Minnesota 55113			8. PERFORMING ORGANIZATION REPORT NUMBER N/A	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Defense Advanced Research Projects Agency Air Force Research Laboratory/IFGB 3701 North Fairfax Drive 525 Brooks Road Arlington VA 22203-1714 Rome New York 13441-4505			10. SPONSORING/MONITORING AGENCY REPORT NUMBER AFRL-IF-RS-TR-2001-204	
11. SUPPLEMENTARY NOTES Air Force Research Laboratory Project Engineer: Mary L. Denz/IFGB/(315) 330-2030				
12a. DISTRIBUTION AVAILABILITY STATEMENT APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) Information Assurance Technologies for the Global Command and Control System (GCCS) Leading Edge Services (LES) program was sponsored by DARPA's Information Systems Office. This report describes the different technology areas the program encompassed, summarized the major achievements of the program, and documents lessons learned and open issues. The technology areas were: (1) Security Architecture. The intent was to provide support for the transition of DARPA technology to operational users and on developing a system security and adversary model that could be used for architectural analysis of the system and information warfare simulations. (2) Distributed Object Security. The primary focus of the work was on CORBA related security with a goal to develop an integrated approach for enhancing the security of a CORBA system. To this end, a proxy for passing the CORBA network protocol, IIOP, through a firewall was developed and implemented on the Sidewinder firewall, and access control mechanisms were implemented to provide security checks on invocation of CORBA methods. (3) Single Sign-on Identification and Authentication. The goal was to develop a single sign-on authentication solution that would eliminate the need to multiple logins when using different applications. (4) Role Based Access Control (RBAC). This technology area used to RBAC model to provide a unified high-level view of the system for administering a system's security policy that would hide the details of the heterogeneous low-level policy and enforcement mechanisms from the security administrator.				
14. SUBJECT TERMS CORBA, Single Sign-On Identification and Authentication, Distributed Object Security, Role Based Access Control, Security Architecture			15. NUMBER OF PAGES 40	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL	

TABLE OF CONTENTS

1	OVERVIEW.....	1
1.1	INTRODUCTION	1
1.2	ACCOMPLISHMENTS.....	2
1.3	DOCUMENT ORGANIZATION.....	3
2	TECHNOLOGY OVERVIEWS.....	5
2.1	SECURITY ARCHITECTURE	5
2.1.1	<i>Checkmate</i>	5
2.1.2	<i>Consulting</i>	10
2.2	ROLE BASED ACCESS CONTROL	10
2.2.1	<i>Napoleon</i>	10
2.2.2	<i>Future Directions</i>	13
2.2.3	<i>Lessons Learned</i>	13
2.2.4	<i>Policy Workshop</i>	14
2.3	IDENTIFICATION AND AUTHENTICATION	16
2.3.1	<i>Single Sign-on Agent</i>	17
2.3.2	<i>Single Sign-on Web Plugin</i>	18
2.4	DISTRIBUTED OBJECT SECURITY	21
2.4.1	<i>IIOP Proxy</i>	21
2.4.2	<i>CORBA Access Control</i>	23
2.4.3	<i>DCOM Study</i>	25
3	SUMMARY.....	28
4	REFERENCES.....	29

TABLE OF FIGURES

Figure 1. The Checkmate Architecture	6
Figure 2. Checkmate Simulation Environment.....	8
Figure 3. The Napoleon Design Tool.....	11
Figure 4. Policy Framework.....	15
Figure 5. Windows Single Sign-on Agent	17
Figure 6. Single Sign-on Web Plugin Architecture	19
Figure 7. IIOP Proxy Configuration.....	22
Figure 8. CORBA Access Control Using Interceptors	24

1 Overview

This document is the final report for the Information Assurance Technologies for the Global Command and Control System (GCCS) Leading Edge Services (LES) program (hereafter referred to as the IAT program) sponsored by DARPA's Information Systems Office. It describes the different areas that the program encompassed, summarizes the major achievements of the program, and documents lessons learned and open issues.

1.1 Introduction

This section contains a brief description of the technology areas that the program covered. More detailed overviews of the technical work are contained in Section 2.

The IAT program was an umbrella program that encompassed four different areas in the information assurance field. These areas were:

- **Security Architecture.** This work was originally intended to be a consulting role on the development of the GCCS LES security architecture. As the IA program developed, however, the focus changed to providing support for the transition of DARPA technology to operational users and on developing a system security and adversary model that could be used for architectural analysis of the system and information warfare simulations. Under this task, Secure Computing participated in DARPA's "PI at the Front" program and various DARPA IA workshops. We also developed the Checkmate network security model and prototyped the Checkmate network security modeling tool that provides a security simulation and analysis capability.[1][2][3]
- **Distributed Object Security.** The objective of the distributed object security work was to identify the gaps between the security that commercial distributed architectures provide and the security requirements that DARPA is trying to address for its operational customers, and then to attempt to address these gaps via IA technology. The primary focus of the work was on CORBA related security, and our goal was to develop an integrated approach for enhancing the security of a CORBA system. To this end, a proxy for passing the CORBA network protocol, IIOP, through a firewall was developed and implemented on the Sidewinder firewall, and access control mechanisms were implemented to provide security checks on invocation of CORBA methods.[12][13] A study on DCOM security was also performed and recommendations made on what might be done to enhance it.[11]
- **Single Sign-on Identification and Authentication.** The single sign-on I&A goal was to develop a single sign-on authentication solution that would eliminate the need for multiple logins when using different applications. An initial prototype was developed for Windows systems that could be used when ftping or telnetting to another system. The prototype showed that the approach of transparently intercepting application login requests and performing the login for the user was feasible but was very application dependent. Having proven the principle of the

approach, the decision was made to focus the remaining work on providing a single sign-on approach for web-based applications that would work across multiple web servers in a domain. A web plugin was developed that would perform the initial authentication of a user and create a cookie for use in automatically logging on to other web servers in the same domain.[10] Secure Computing's commercial division eventually adapted the plugin for use with its SafeWord authentication server.

- Role Based Access Control (RBAC). In this area, the objective was to use the RBAC model to provide a unified high-level view of the system for administering a system's security policy that would hide the details of the heterogeneous low-level policy and enforcement mechanisms from the security administrator. A model and tool was developed, called Napoleon, that presented a layered view of the various components of the system and allowed roles to be defined that spanned a wide variety of enforcement mechanisms.[4][5] Napoleon also provides a translation capability that can map high-level role based policies to the low-level mechanism specific policies that are needed to enforce the high-level policy. This allows a system administrator to maintain system policies at a high-level without worrying about how the policies should be represented on specific enforcement mechanisms. A number of papers describing Napoleon were written and presented at various conferences.[6][7][8][9]

In addition, Secure Computing supported the integration of our work into experiments at the IA Technology Integration Center (TIC). In particular, as part of this integration support, Secure Computing ported Boeing's IDIP code to the Sidewinder firewall.

1.2 Accomplishments

On this program, the following specific accomplishments were achieved.

- Secure Computing developed a comprehensive system security model that includes models of the network topology and network components, mission objectives and adversary characteristics, and a database of component vulnerabilities. The model can be used as the basis for Course of Action analysis tools and Intrusion Detection and Response simulations.
- A network security analysis tool, called Checkmate, that is based on the model was prototyped and evaluated. The final prototype allows modeling of a useful subset of the services typically provided by a distributed system. The prototype allows a live attacker to selectively attack a static (configured at initialization) system model and determine the impact on system services. Checkmate is currently being enhanced on other programs to provide a network security simulation environment.
- A paper describing the Checkmate model and security analysis tool was prepared and presented at Milcom 2000[3].
- A paper suggesting some possible new modeling paradigms for COTS based

command and control systems was prepared and presented at the 1999 New Security Paradigms Workshop[15].

- Secure Computing develop the Napoleon Role Based Access Control (RBAC) model for unifying security policy management across heterogeneous network components. The model supports semantic layers, such as wrappers, that combine multiple mechanisms and translation components.
- The Napoleon tool, based on the model, was developed to support easy definition of consistent local and system-wide RBAC policies in distributed object architectures and then to automatically map these policies to enforcement mechanisms. Mappings were developed for CORBA, DTEL++, DCOM and NAI Labs data-driven generic wrappers.
- Secure Computing performed a study that investigated how the Napoleon RBAC approach could be integrated with the CORBASEC V2 Rights functionality.
- Papers on the Napoleon work were prepared and presented at ACSAC98[6], the 1999 ACM RBAC Conference[7], ACSAC99[8], and Milcom2000[9].
- Secure Computing developed a prototype IIOP proxy on our Sidewinder firewall. A joint whitepaper was written with NAI Labs on issues in proxying IIOP traffic through a firewall.
- Secure Computing developed and prototyped an interceptor based approach for adding object based access control to CORBA applications. The interceptor technology was integrated with the Open Group's Pledge policy management system and with Napoleon.
- Secure Computing performed a DCOM security study that reviewed the DCOM security approach, identified gaps in the approach and suggested possible solutions using IA technologies. Use of a DCOM interceptor for fine-grained access control was prototyped.
- Secure Computing developed a prototype of an agent based single sign-on system for Windows NT.
- Secure Computing developed a web plugin that supports single sign-on. The plugin technology was transitioned to our commercial products.
- Secure Computing supported other DARPA programs in a number of ways including integration of Boeing's IDIP technology into our Sidewinder firewall and integration of the Napoleon RBAC policy management tool with Network Associates' DTE and generic wrapper technologies. Tom Markham also acted as an IA PI-at-the-front in support of DARPA-PACOM cooperative activities.

1.3 Document Organization

The remainder of this document is organized as follows.

- Section 2 contains a more detailed description of the various technology areas

including the advances made, possible future directions and lessons learned.

- Section 3 presents a short summary of the program.
- Section 4 includes a list of cited and program-produced documentation.

2 Technology Overviews

In this section each of the four different areas of technology development identified in Section 1.1 is discussed in more detail.

2.1 Security Architecture

In the security architecture area, Secure Computing developed the prototype Checkmate network security modeling tool and supported DARPA's technology transition efforts.

2.1.1 Checkmate

Effective reasoning about system attacks and responses requires a comprehensive model that covers all aspects of the system being analyzed, from network topology and configuration, to specific vulnerabilities, to possible adversary capabilities and possible attacks. A comprehensive model can be used as the basis for real-time attack/response simulations, "what if" course of action analysis, policy simulation and debugging, and more. To address this need, Secure Computing developed a system called Checkmate, a prototype network security model and tool for representing and investigating the security of real-world networks.

The Checkmate model represents the topology and configuration of network components such as servers, workstations, routers, and firewalls. The model also includes potential network operations (including vulnerabilities), mission-critical files and services supported by the network, and characteristics of potential attackers and defenders in a manner that allows security analysis to be performed.

The model represents network operations (including vulnerabilities) at the service level with an abstraction of each service's functionality. Each operation has a set of preconditions and a set of effects. Examples of preconditions include the operating system and service versions to which the operation applies and client resources required by the operation. The effects of an operation can include changing the state of a node or protocol, routing to a new node or changing to a new protocol, triggering alarms, and discovering resources that may be used as parameters in future attack actions. When applied to a node, some operations fail because of defenses, proper configuration or other security policy implementation. Some of the operations perform normal network functions. Many of the operations exploit weaknesses that have the effects described in attack and vulnerability databases. These effects may change the state of the system, removing the defenses against attacks on other services.

The Checkmate tool simulates attacker and defender interactions using the Checkmate model to represent the network, operations, and missions. The tool provides a state-space representation of the network under test. A particular network can be analyzed by investigating possible attacks on the network based on the vulnerabilities in the model.

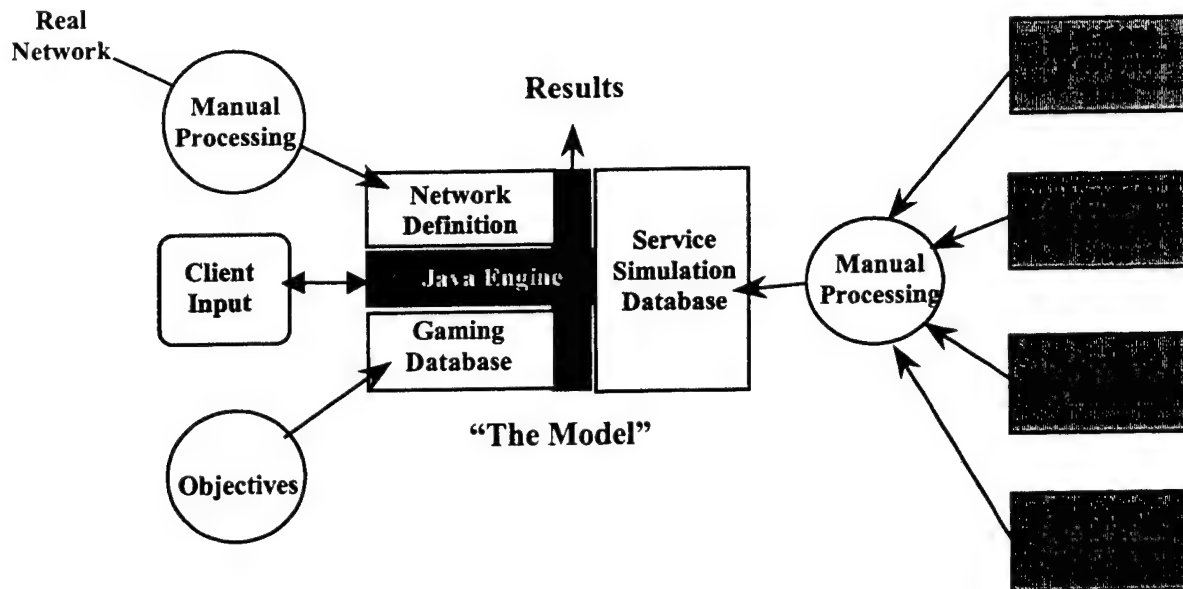


Figure 1. The Checkmate Architecture

With the Checkmate tool, an attacker attempts to gain access to components in the simulated network, extract or alter confidential information, and shut down critical services without raising alarms that would indicate the attacker's presence. In order for the attacker to execute a particular exploit, it may be necessary for the attacker to execute vulnerabilities in several protocols on different systems.

A defender attempts to block an attacker's actions while maintaining the availability, security, and integrity of mission critical data and services. The defender does this by monitoring alarms and appropriately modifying the defense parameters of nodes in the simulated network. Characteristics of a node's configuration that a defender can modify include operating system brand and version installed, services and versions installed, service-specific settings (e.g. does the FTP server allow writes), password strengths and access levels, and firewall or router filtering rules.

In its interactive mode, the Checkmate tool provides a vehicle for information war gaming. Adversaries attack the system and defenders dynamically adjust defenses to counter the attack. Because Checkmate uses actual network configurations and vulnerabilities, such war gaming could also be used to train defensive and offensive information warriors.

The architecture of the prototype, illustrated in Figure 1, is client-server based. The Checkmate server maintains the network model, service and vulnerability database, mission objectives, and current system state. The system is modeled at the service

level (e.g., **ftp**, **rlogin**, and **snmp**) with an abstraction of each service's functionality as described in the system's help manuals. Specific service vulnerabilities and the service versions that contain the vulnerability are part of the model, and are obtained from a variety of vulnerability and attack databases, such as CERT[16], Bugtraq[17], and rootshell[18]. The level of the model is designed to contain enough information to be a useful representation without being overly detailed.

The simulation works as follows. Using an attack script generated off-line, an automated attacker client submits the sequence of actions in the script to the Checkmate server for processing. The actions could be timed in such a manner as to represent an actual automated attack, for example, one action per second. The server evaluates the attack action and applies the effects of that action to the modeled network.

The autonomic response tool acts as an automated defender client. It receives alarms that are triggered by the attack from the Checkmate server. In response to specific alarms, it may dynamically modify the defense configurations of individual nodes. Attack actions are processed in the order they appear in the attack script. By monitoring alarms and appropriately modifying the defense configurations of nodes in the modeled network, the defender client can block an attacker client. The attacker client continues to process the attack script until an action fails.

2.1.1.1 Future Directions

Currently, Checkmate development is continuing in two areas:

- On a subcontract to Honeywell Technology Center's (HTC) CIRCADIA program, sponsored by DARPA, Checkmate is being enhanced to provide a network simulation environment for testing autonomic intrusion response tools. The approach is described below.
- Lawrence Livermore National Laboratory (LLNL) is funding Secure Computing to integrate Checkmate into their IOWA environment to provide a simulation environment.

Some of the possible future applications for the Checkmate technology are discussed below.

Simulations

The Checkmate tool could be used to provide a simulation capability for testing and evaluating intrusion response tools, among other things. The simulation environment is illustrated Figure 2.

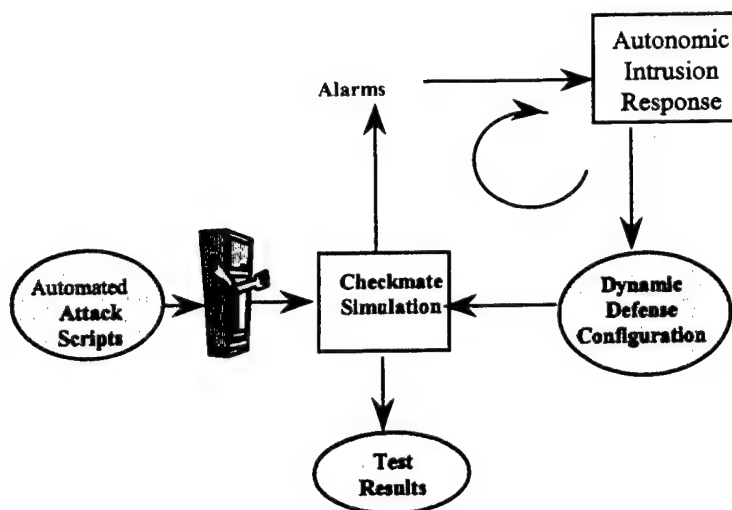


Figure 2. Checkmate Simulation Environment

The resulting simulation environment could be used to measure the effectiveness of intrusion response tools in providing an autonomic response to detected intrusions. The coverage that the response tool provides could be measured by automating a wide variety of attacks and determining which ones the tool is capable of detecting and deflecting. The real-time nature of the response could be measured by running the attack scripts at different speeds and then determining if there is a point at which the attack is coming too fast for the tool to effectively respond.

Course of Action Analysis

The Checkmate tool could be used as a course of action tool to provide feedback on the implications of a particular system being compromised, to help predict which systems might be attacked next, to suggest possible responses, and to analyze the overall implications of a particular response. Checkmate would be able to provide “what if” analysis that could identify the security ramifications of installing a new service or policy, or the consequences of a particular system being compromised.

Policy Simulation and Debugging

A policy simulation and debugging capability could be developed using Checkmate. Policy definition tools, such as Napoleon, could be used to define and then translate the policies to the Checkmate model. Simulations could check things such as: connectivity for certain applications, availability, and why a requested access failed.

Game Playing and Training

In its interactive mode, Checkmate provides a tool for information war gaming. Adversaries attack the system and defenders dynamically adjust defenses to counter the attack. Since Checkmate uses actual network configurations and vulnerabilities, such war gaming could also be used to train defensive and offensive information

warriors.

2.1.1.2 Lessons Learned

The original objective of the Checkmate work was to determine whether a network could be modeled in enough detail to be able to perform meaningful security analysis of the network by using the model. The success of the Checkmate prototype, and its use by HTC and LLNL, indicates the feasibility of the approach. However, a number of issues have been identified that would need to be addressed before Checkmate could be a general purpose analysis tool. Some of these are:

Network Discovery

An evaluation of the Checkmate tool was performed on the IAT program to determine its usability and effectiveness. As part of this evaluation, the network that was being used for the DARPA IA Integrated Feasibility Experiment(IFE) 2.3 was represented in the Checkmate model. Since Checkmate does not support automated network discovery, entering the data into the model was an extremely labor-intensive effort. Clearly, if Checkmate is to be used on a wider basis, some form of automatic network discovery that could populate the model is required. There are a number of commercial and research tools that could help solve this problem, and some of our work with LLNL is also addressing it, but the amount and extent of the information needed by the Checkmate model might require an agent based, information gathering approach.

The Vulnerability Database

Checkmate's analysis depends on the timeliness and completeness of the vulnerability database that it uses. This vulnerability database is relatively detailed and includes actions that might result from the vulnerability being exploited as well as defenses against the vulnerability. Currently, the only way to populate the database is via a manual process that involves scanning well-known sources of vulnerabilities and attacks, such as CERT, bugtraq and rootshell, extracting from them the information needed, and coding it in a form that Checkmate understands. Although each vulnerability only needs to be categorized once in this manner, it is still a time-consuming process and makes it difficult to maintain the currency of the database.

To solve this problem, it would be useful for the security community to develop a comprehensive vulnerability classification scheme that includes all of the information that Checkmate requires in a machine processable format. While some efforts are underway in this area, such as MITRE's CVE work, they are only modest beginnings on what really is necessary to allow tools to be built that could easily import vulnerability databases and reason using them.

In addition, to provide more significant analysis capability, an attack signature database is necessary. This database would model complex attacks based upon characteristic network configurations and upon characteristic events observed during the progress of the attacks. Attack signatures could be developed from well-known

attack, exploit, and vulnerability descriptions, full-disclosure discussion lists and Web sites, commercial vendor information, and interviews with expert practitioners in the field of information warfare. Using attack signatures rather than specific vulnerabilities would allow generic attacks to be identified that work if certain easily-identified preconditions hold.

2.1.2 Consulting

In the consulting area of the security architecture task, Secure Computing participated in a variety of IA workshops and activities, was actively involved in the "PI at the front" effort, and wrote or co-wrote papers including:

- An architecture and design document for composable identification/authentication and authorization services. This was done as part of the initial composable services effort of the IA program.
- A paper on security modeling, co-authored with Mary Denz of AFRL, that was presented at the New Security Paradigms Workshop in 1999.[14] The objective of the paper was to stimulate discussion on the future of security modeling. It asserted that top down security modeling is non-existent today and suggested some possible modeling paradigms for COTS based command and control systems.

2.2 Role Based Access Control

In the RBAC area, Secure Computing developed the Napoleon RBAC policy management tool and initiated the first IA Policy Workshop.

2.2.1 Napoleon

Napoleon is the name of the framework and set of RBAC tools for management of local and system wide security policies that Secure Computing developed under the IAT program[4] [5]. The effort emphasized a practical approach that does not require existing applications to be rewritten so that they can be incorporated into the framework. The result is a security administration umbrella that incorporates a wide variety of access control mechanisms.

The general RBAC concept involves:

- defining the set of roles for the system. A role is a set of permissions needed to perform some job function. For example, what access does a mission planner need to a database? What access does the flight crew need to the database? These permissions are bundled together into the mission planner role and the flight crew role.
- defining the objects to which each role has access and the particular operations that someone in the role can perform on each object
- associating each user with the roles they can perform on the system

- associating a current role with each user that identifies in which role the user is currently operating.

A user can only access an object if the permissions in their currently active role allow it.

RBAC policies are currently receiving much attention, and mechanisms that can be used to support such policies, either directly as in database roles or indirectly as in CORBA rights, are being developed and deployed commercially. The goal of this effort was to develop a framework and a tool that allows all of the different types of RBAC systems to be incorporated into a unified environment.

The Napoleon approach is illustrated in Figure 3. Napoleon has two parts:

- a GUI design tool that allows the administrator to create roles in a natural way by using the Napoleon policy layer model,
- a set of policy translators that map roles from the model into the legacy security mechanisms that actually enforce the policy.

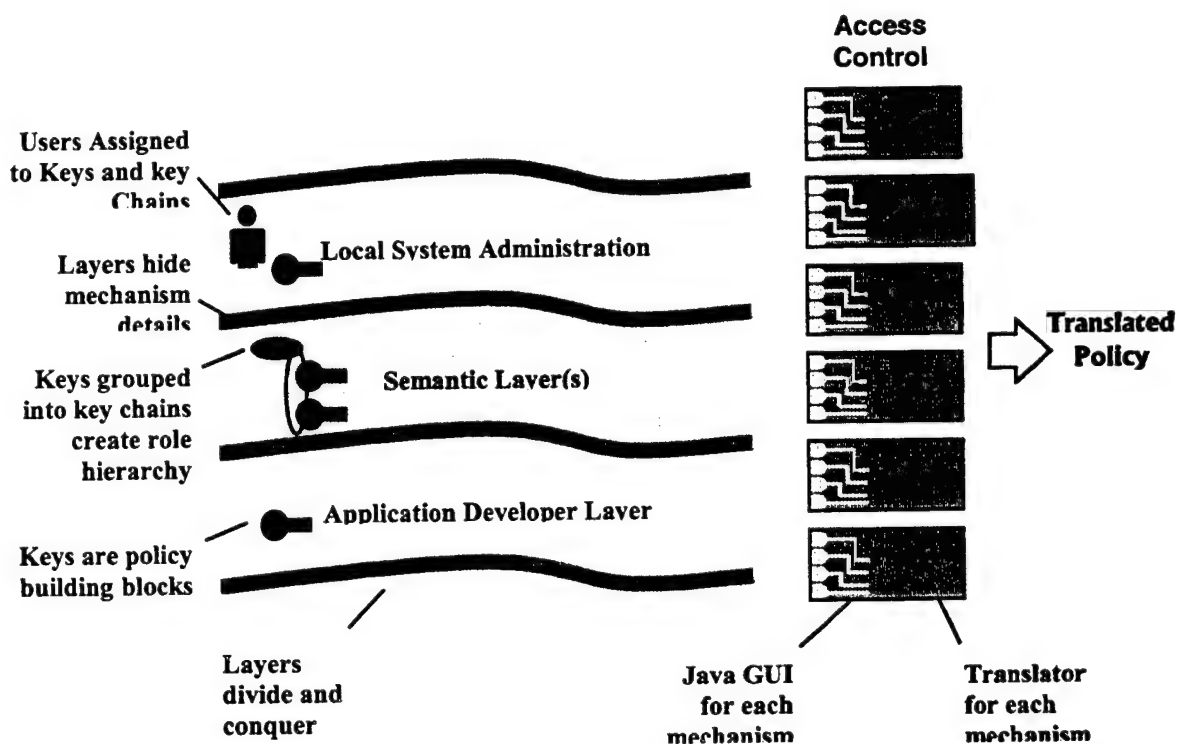


Figure 3. The Napoleon Design Tool

The Napoleon approach provides the following advantages:

- roles can be defined for an entire network of diverse mechanisms and applications using a single, simple tool, without worrying about how the roles are implemented on the various systems
- role translators that map the policy to a specific enforcement mechanism hide the implementation details from security administrators
- the role translators can be analyzed to ensure they perform the mapping from role definitions to role enforcement mechanisms in a correct manner
- adding a new enforcement mechanism only requires creating a role translator
- an intuitive interface to the role concept, keys and key chains, is provided
- a distinct design layer is used to provide structure to the complicated task of designing security policy.

To make the abstract concept of a role more concrete, the Napoleon design tool uses a concept understandable by all users, the key. In the real world if a person possesses a key to a door, they can access all the resources behind the door. In Napoleon if a user is given a key they have access to the resources represented by that key. For example, a Napoleon key may represent read access to a set of files, or the ability to update a database table.

In the real world the set of resources to be protected is small. Usually a person needs less than ten keys. On a computer network the number of resources to be controlled is staggering. A file system alone may contain 10,000 different files that need to be controlled. If the system administrator had to assign one key to a user for each file the user could access, the job would never be done.

Napoleon provides two concepts to aid in scaling up to control a larger set of resources. The first is the key chain. The key chain is simply a collection of keys. If a user is given a key chain, they can access all the resources associated with all the keys on the key chain. As in the real world, a key chain may contain other key chains. For example, all your work related keys may be on one ring, and your home keys, on another. In Napoleon key chains are used to group related keys into convenient sets. For example, several file keys relating to mission planning could be combined into a single key chain. Now the new "mission planning" key chain could be given out to users in one operation.

Policy layers are the second concept that increases scalability. Policy layers organize keys and key chains into related areas. For example, all the mission planning keys and key chains that control database resources are in a single layer. Each application or mechanism gets its own layer. The major benefit of layers is that it allows the application developer to group application resources into convenient sets. As a result an application can be installed on the network with some of the policy already built, saving the system administrator a great deal of effort. Layers are a design paradigm that greatly simplify the creation of security policies.

The Napoleon design tool presents a standardized, intuitive version of roles to the policy developer. All details on how the roles are actually implemented within applications are hidden by the layer feature. The policy designer combines keys and key chains to build policy. The Napoleon design tool saves the policy as an XML file.

The role translators take the XML file and generate an appropriate representation of the policy for the target enforcement mechanism. The enforcement mechanism might require new configuration files or an administrative command to change the policy. Secure Computing has built role translators for several enforcement mechanisms:

- the LOCK6 decision server that SCC created to enforce fine grained policies for the CORBA environment using CORBA interceptors
- the Adage/Pledge decision server that the Open Group Research Institute (OGRI) developed[21] and SCC integrated into the CORBA environment using CORBA interceptors
- the DTEL environment for enforcing fine grained CORBA policies developed by NAI labs[19], and
- the data driven process wrappers developed by NAI labs[20].

To summarize, Napoleon is an RBAC solution to policy management. Napoleon provides several unique features such as key and key chains and policy layers to provide structure and ease the challenging task of managing security policy.

2.2.2 Future Directions

Napoleon development is continuing on a Phase II SBIR from NIST. On that program the Napoleon model is being enhanced to include workflow policies and to handle some simple constraints. In addition, a number of Napoleon concepts have been transitioned to other programs, such as DARPA's Autonomic Distributed Firewall and AFRL's Kernel Loadable Wrappers, that require policy management.

2.2.3 Lessons Learned

The Napoleon work demonstrated that role based access control is a viable approach for unifying the policy management of various heterogeneous enforcement mechanisms. It also validated the layered approach to specifying policy.

User testing was performed to help pinpoint areas where future improvements could be done. A number of issues with the current tool were identified, such as the need to specifically import keys from lower layers before they could be used at a higher layer and the need to create key chains out of keys before they could be imported. Many of these issues are being addressed on the NIST program.

While the Napoleon model included the notion of constraints to add granularity to a policy and allow specific conditions, such as the time of day or an attribute value of a particular object, to be included in the policy, these constraints were never

implemented in the Napoleon tool. The issue with constraints is not in specifying them using Napoleon (this could be easily done), but is primarily with mapping the constraints to appropriate enforcement mechanisms. For some types of constraints, such as date or time of day, there may be enforcement mechanisms that could be used to enforce the constraint, but for many others, such as those that depend on an attribute value of an object, no current enforcement mechanisms support enforcement of the constraints. Hence, without creating one's own enforcement mechanism, these types of constraints are not practically useful. In summary, the lesson learned is that specifying policy constraints is of no use unless there are enforcement mechanisms that are capable of enforcing the constraints.

2.2.4 Policy Workshop

As part of the RBAC task, Secure Computing organized a Policy Workshop that was held at BBN in May 1998. Other participants were BBN, Open Group Research Institute and NAI Labs (at that time, TIS).

The major technical goal of the workshop was to:

- Develop an IA program-wide Policy Server Architecture/Framework that includes:
 - policy definition and management for a wide variety of policies
 - policy enforcement in both CORBA and non-CORBA environments
 - a uniform user interface.

An additional programmatic goal was to:

- Develop a plan for implementing the Policy Server Architecture/Framework that:
 - builds on and integrates the work that programs have already done
 - avoids duplication of effort across programs
 - provides each program the opportunity to make a meaningful contribution to the overall architecture/framework.

This section will summarize the technical results of the workshop only.

Results

A major result of the workshop was the definition of a preliminary policy architecture/framework that could provide a common vocabulary for use in future policy discussions on the IA program. The architecture is shown in Figure 3.

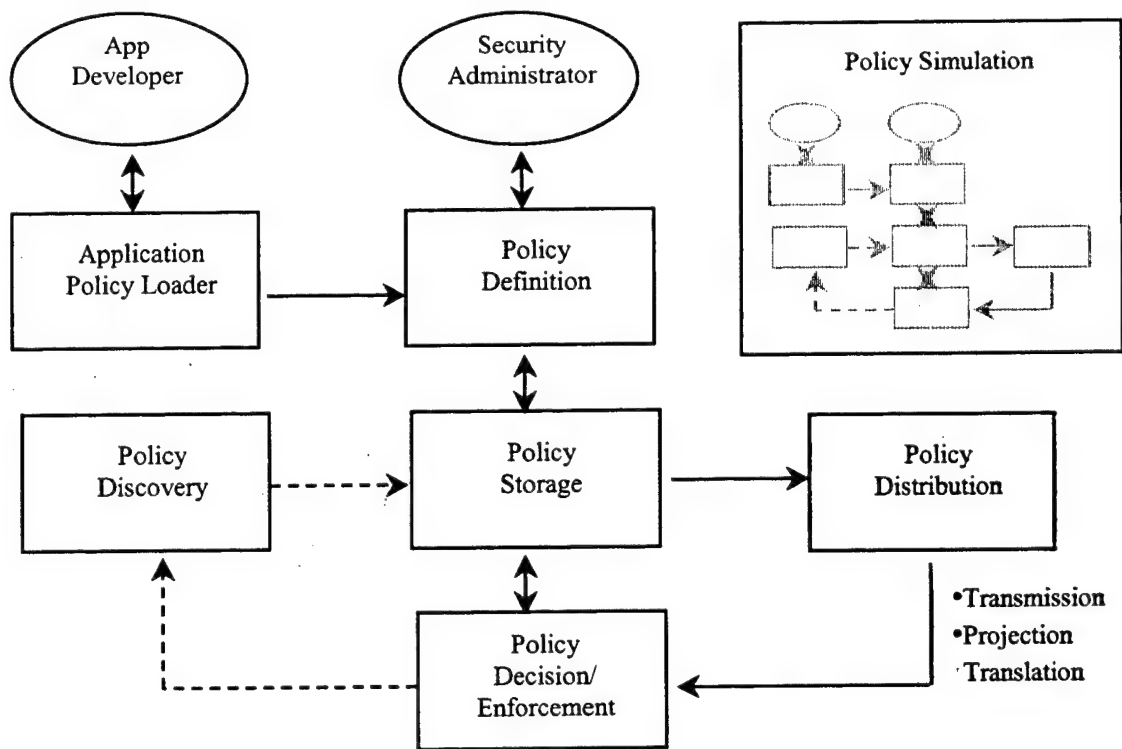


Figure 4. Policy Framework

Components of the Policy Architecture/Framework:

1. **Policy Definition:** handles creating, modifying and deleting policies and policy elements. This area provides the user interface for manipulating policies.
2. **Policy Storage:** handles persistent, centralized storage of the high level policy according to a well-defined policy schema. Policy storage could be done via a DBMS.
3. **Policy Distribution:** handles transmission, projection and translation of the high level policy, as stored in the policy storage component, to the policy enforcement mechanisms. Transmission refers to the means by which the policy is communicated to the enforcement mechanisms; projection refers to the process of identifying what portions of the high-level policy are the responsibility of each particular enforcement mechanism; and translation refers to the process of mapping the projected policy to the particular form used by the enforcement mechanism to represent the policy it enforces.
4. **Policy Decision/Enforcement:** handles run-time policy decisions and mechanism specific enforcement. This area includes storage of the translated policy in an

appropriate low-level format for rapid decision making. As in the policy distribution area, there are a variety of decision/enforcement mechanisms that must be supported, including application component frameworks such as CORBA, COM and Enterprise Java Beans, operating systems, network components such as firewalls and filtering routers, authentication systems, and intrusion detection systems.

5. Application Policy Definition/Loader: provides the means for application developers/analysts to define policy components as part of their application and for loading external policy elements into the policy storage.
6. Policy Discovery: provides the ability to identify low-level policies that are actually being enforced and compare these policies to the high-level policy statements to check for consistency.
7. Policy Simulation: provides the ability to simulate changes to the policy and determine their effect on the overall security of the system. Policy simulation is an additional component that is needed but is not explicitly a part of the policy server architecture/framework. Such a capability could be used by planners who need to modify policy as part of a re-plan and would like to understand better what the policy modifications imply.

Scope

The policy server architecture/framework described in the report focuses on solving the policy management and enforcement problem within a single policy domain controlled by a central Command and Control desk. Extending the work across multiple policy domains is not addressed.

A variety of policies can be supported by the framework including authentication, authorization, confidentiality, integrity, quality of service, and intrusion/detection. While the framework supports flexible policies, the amount of flexibility needed to, for example, express constraints such as time, location or previous accesses, is not yet fully known and additional real-world examples will be needed to clarify this. The framework is aimed at supporting a wide range of applications and systems, both new and legacy: including distributed applications based on CORBA, COM, Java RMI, and the web; email applications; operating systems such as NT and Solaris; routers and firewalls; authentication servers; dial-in servers; and others.

2.3 Identification and Authentication

The objective of the Identification and Authentication (I&A) task of the IAT project was to develop a single sign-on approach to address the need for multiple different passwords for different applications. Work was performed in two areas:

- Development of a single sign-on agent on a Windows NT system
- Development of a single sign-on plugin for a web browser.

Each is discussed in more detail below.

2.3.1 Single Sign-on Agent

The original I&A work was focused on developing a single sign-on approach that could be used on a Windows system for standard applications. The approach chosen was to use a single sign-on agent that the user would authenticate to once via strong authentication using tokens. The agent would then handle all future logins.

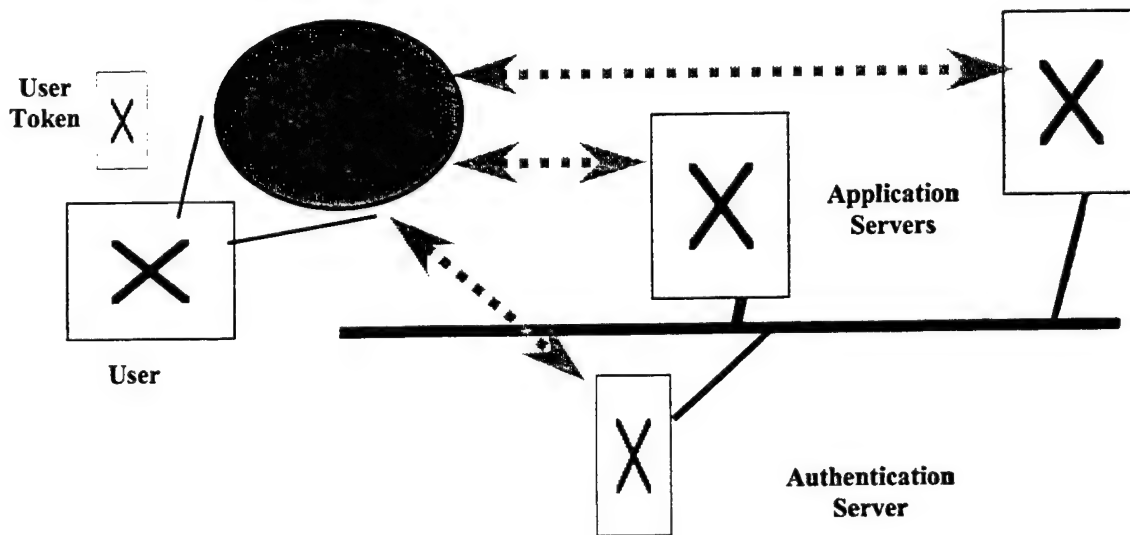


Figure 5. Windows Single Sign-on Agent

An agent was developed for Windows NT that provided a single sign-on capability for telnet or ftp to other systems. Initially the user would login to the agent to unlock the user passwords and/or pins that the agent had stored. This login was performed using the RADIUS authentication standard to communicate with an authentication server. The authentication could be hardware based or password based. The agent built up its password list automatically by monitoring logins that the user had done previously and capturing the server, application and password. Once initialized, the agent would intercept the telnet and ftp login requests that were sent by a server and then would respond with the correct password. The agent included a software based authentication token so that one-time passwords could be computed automatically. A demonstration of the system was given at the Integrated Feasibility Demo 1.2.

2.3.1.1 Future Directions

As discussed in the Lessons Learned section below, the approach relies on the agent understanding portions of any protocol for which it is providing a single sign-on capability. After developing the telnet and ftp single sign-on capabilities, it was decided that adding other protocols, one at a time, was not a cost-effective approach to the problem, and that a more useful focus for the task would be on providing a

single sign-on capability for web based applications. Hence, no further work was done on the prototype and our work effort shifted to the web based plugin discussed in Section 2.3.2 below.

2.3.1.2 Lessons Learned

The approach relied on intercepting the network traffic at the protocol level and recognizing from the network traffic when an authentication request had been received from a server. This implies that the agent understands the protocol being used well enough so that it can interpret the protocol traffic to identify the authentication request. Every protocol has its own way of making an authentication request; hence adding a new protocol to those that the agent understands involves adding another protocol specific piece of code. While this is doable for most standardized protocols, it can involve a large amount of effort.

An approach was developed, but not implemented, that would use a general algorithm for capturing and responding to authentication requests from various protocols. This approach would require specifying some key portions of a new protocol, such as keywords that identify the authentication request and the format for a response, and then would use a common portion of code to intercept the request and respond in the appropriate format. Potentially, this could make adding a new protocol to the agent much easier. However, there will always be some protocol specific code that must be generated, and for complex protocols, this may be a significant task.

An additional difficulty with the approach lies in the fact that many application protocols are proprietary and, hence, details of the protocol needed by the agent are not available. This implies that applications that use proprietary protocols, such as SQLNet from Oracle, cannot be handled easily with the approach.

2.3.2 Single Sign-on Web Plugin

The single sign-on web plugin was developed to eliminate the need for a user to login to each individual web server in a domain (such as darpa.mil) when the user needed to retrieve information from multiple servers in the same domain[10].

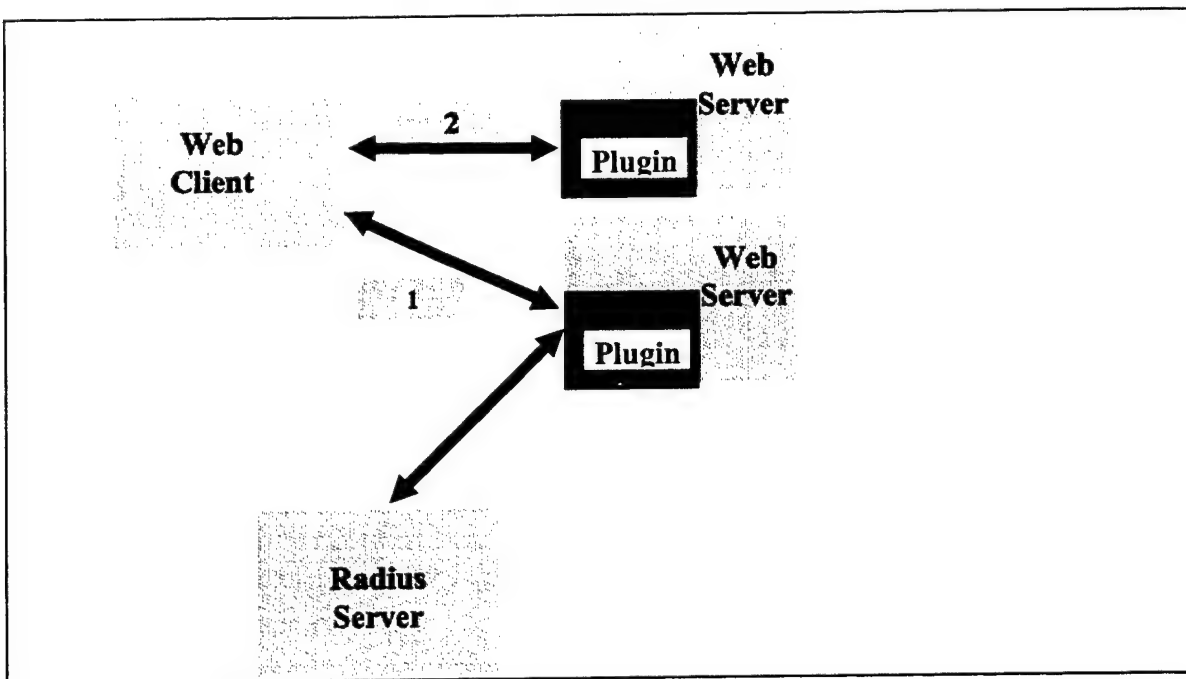


Figure 6. Single Sign-on Web Plugin Architecture

Figure 6 illustrates the plugin architecture.

The plugin intercepts all web requests and can perform authentication and role-based authorization. The plugin first asks for the user name and then queries the Radius Server for the authentication requirements for the user. Based on the authentication requirements for the user, the plugin generates a customized form for entering the required authentication information and returns it to the user. For example, the form may ask for a fixed password as well as a synchronous password.

When the user completes the authentication form, the plugin passes the information to the authentication server for verification. If the verification is successful, the authentication server returns the list of roles, to which the user is allowed, back to the plugin. The plugin then generates a cookie that is used to allow reconnections for a period of time without having to re-authenticate to the authentication server. The cookie is encrypted and contains information on the user such as to what roles the user is permitted and when the user authenticated himself.

On any subsequent request for pages from any web server in the same domain, the browser automatically inserts the cookie. If the web server is running a plugin, then the cookie is decrypted by the plugin and checked to see that the user authentication has not expired and that the user is in a role that is allowed to see the web page requested. In particular, single sign-on among web servers in the same domain is supported in this manner.

The connection 1 (or 2) between the client and server is either http, or http over SSL. In the case of SSL, the connection can be either server authenticated (the most common use of SSL currently) or mutually authenticated (if the client has a certificate and the server requires mutual authentication). For a mutually authenticated connection, the plugin can pull out the client certificate and specific information in the certificate and pass this on to the authentication server for additional verification.

For a Netscape web server, role based authorization to specific web pages is enforced by the plugin using the roles that are contained in the user's cookie and the mapping of roles to specific web pages that must be entered into the Netscape obj.conf configuration file. A feasibility demonstration to automatically create the obj.conf file was done using Napoleon, however, there currently is no other tool for modifying the Netscape obj.conf file; i.e. it must be edited by hand. If all files on the web server are either accessible or not based on the user's role, then this is not hard. Once you start adding more fine-grained access, this can get fairly complicated.

Usage Scenarios

1. No SSL on connection 1. The user could be required to authenticate before getting access to the Web Server. This scenario probably has limited usefulness since connection 1 is not encrypted.
2. Server authenticated SSL on connection 1. As in case 1, the client could be required to authenticate to the server. In the short-term, this may be a very useful scenario, since this form of SSL is what is currently dominant on the Web. Most users will not have certificates, so user authentication could be done using an authentication server. The plugin could also be used in heterogeneous situations where some users have certificates and others use fixed passwords or tokens.
3. Mutual authentication via SSL. Both client and server have certificates and are authenticated using SSL. The plugin could be used to pull off the client's certificate and use the information in it to check for certificate validity and client authorizations. It could also be used to do additional authentication of the user.

2.3.2.1 Future Directions

The value of the single sign-on web plugin was recognized by Secure Computing's commercial authentication division and that division took the prototype developed under this program and did the productization work necessary to add it as a feature to Secure Computing's SafeWord authentication product. Further development and enhancements will be done as needed by this commercial division. This is a good example of work from the IA program being successfully transitioned to a commercial product.

2.3.2.2 Lessons Learned

The single sign-on web plugin was successful in solving the problem that it was addressing and then was incorporated into a commercial product because the focus of

the work was well-defined and addressed an issue that was a commercial as well as a Department of Defense problem. The solution used techniques that were being developed by the commercial world as opposed to one-off solutions for government work. The lesson to be learned here is that the DoD is currently running on commercial products and so any solutions to specific DoD problems must be in the context of the commercial products and involve approaches that are acceptable in the commercial world. Otherwise, the rapid advance of commercial technology will quickly obsolete any DoD specific solutions.

2.4 Distributed Object Security

The original focus of the distributed object security (DOS) task was on the Common Object Request Broker Architecture (CORBA) and, in particular, developing a series of more sophisticated firewall proxies for CORBA traffic over the Internet Inter-ORB Protocol (IIOP). Because of redundancy in efforts between Secure Computing's proposed work and the work proposed by NAI Labs, the work performed by Secure Computing was redirected to concentrate on CORBA access control and an initial investigation into the security issues associated with Microsoft's DCOM architecture. As a result, there were three main achievements in this area:

- A basic IIOP proxy for Secure Computing's Sidewinder firewall was developed.
- An approach for providing access control to CORBA method invocations was developed.
- A study of DCOM security issues was undertaken and recommendations made on possible uses of IA technology to augment DCOM security.

Each area is discussed in more detail below.

2.4.1 IIOP Proxy

Internet Inter-ORB Protocol (IIOPTM) is the standard protocol employed by CORBA^{TM1} applications to access and invoke operations on objects in a distributed environment using TCP/IP. When the distributed environment includes the Internet, then the IIOP traffic must be able to pass through firewalls that are protecting the enclaves where the application clients and servers run. However, application firewalls will not allow IIOP traffic to pass through the firewall unless it is filtered by an appropriate proxy that understands the IIOP protocol. A basic IIOP proxy for Secure Computing's Sidewinder application firewall was developed on this program. The IIOP Proxy enables fine grained access control to CORBA objects, interfaces and methods using well-known firewall techniques.

The design builds on Iona Technologies² WonderWallTM, an IIOP proxy for OrbixTM environments. The design includes details for integrating WonderWall with the firewall features and operating system protection provided by Secure Computing's

¹ IIOP and CORBA are registered trademarks of the Object Management Group (OMG).

² WonderWall and Orbix are registered trademarks of Iona Technologies, Ltd.

Sidewinder.

In our implementation, WonderWall runs as a type enforced proxy on the Sidewinder. It has been enhanced to include:

- support for transparent access
- support for multiple ports
- support for Visigenics and other non-Orbix servers
- easier configuration.

Access control is based on:

- message type
- source address
- destination address
- object key
- method invoked.

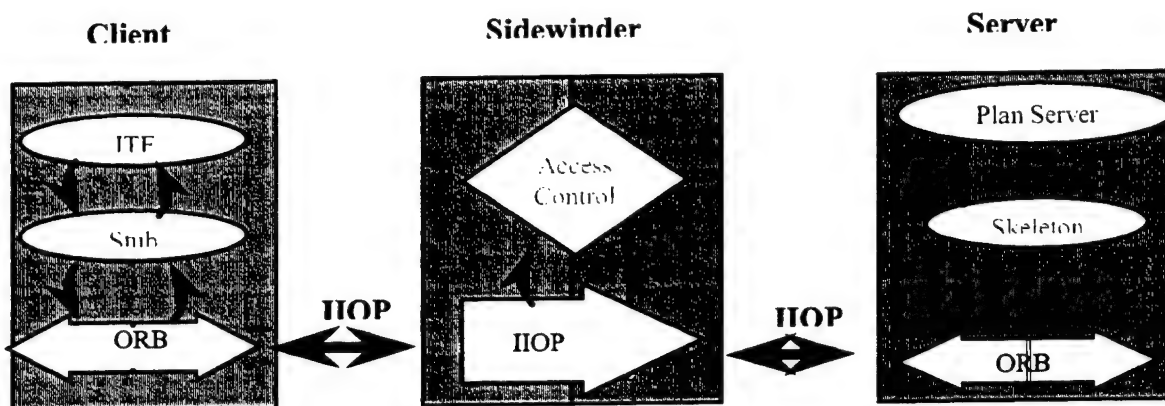


Figure 7. IIOP Proxy Configuration

Figure 7 shows the basic proxy architecture. The proxy policy is configured using a standard Sidewinder configuration file. Once configured, the proxy will only allow IIOP traffic through the Sidewinder that satisfies the IIOP access control rules. The Proxy Design[12] and User Guide[13] describe the system in more detail.

2.4.1.1 Future Directions

Although it was hoped that the IIOP proxy would eventually be transitioned to the commercial division of Secure Computing for inclusion in the Sidewinder product,

there was never a sufficient commercial demand for the proxy that would justify the extra work needed to convert the prototype into a product feature. In the rapidly changing world of the Internet, the IIOP protocol has never caught on as an Internet protocol that would need to be proxied through firewalls. In fact, in many cases the solution that the CORBA vendors are providing uses http to tunnel IIOP through firewalls since all firewalls already have http proxies.

2.4.1.2 Lessons Learned

When building the prototype, there were some specific lessons learned that apply to the proxy technology under investigation. These include:

- Access control at the proxy is limited to control of the object class. In particular, if several object instances of a class exist, the proxy cannot distinguish between them.
- Use of approaches that bundle calls to increase performance (such as the "fine c2" used by the composable services) limits granularity of access control that can be enforced. In these approaches once the calls are bundled, a generic method is called that transmits the bundle from client to server. The only access control that can be performed at the proxy is on the generic method.
- Different vendors use different formats for their object identifiers. This implies that a proxy will have to know a specific vendor format before it can successfully proxy traffic for that vendor's CORBA implementation. This is a problem that the OMG is attempting to address.

Aside from these specific lessons, a more general lesson is that one can never know for certain which technologies will predominate on the Internet when those technologies are first introduced. When this program started, it appeared that IIOP would be the next big Internet protocol and that it would be used instead of http for complex applications. This has not happened, and, as a result, the need for commercial IIOP proxies has been very limited.

2.4.2 CORBA Access Control

The goal of the CORBA Access Control effort was to fill in the gaps between what DARPA needs and COTS implementations provide. While the CORBA Security Specification (CORBASEC) had already been approved by the Object Management Group (OMG), vendors were very slow in implementing the security features and no implementations of level 2 CORBASEC were available when our work was undertaken.

Secure Computing's approach to this problem was based on identifying areas where commercial CORBA implementations provide "hooks" for replaceable components. Elements of the IA architecture would then be applied to fill in security gaps by exploiting these replaceability features.

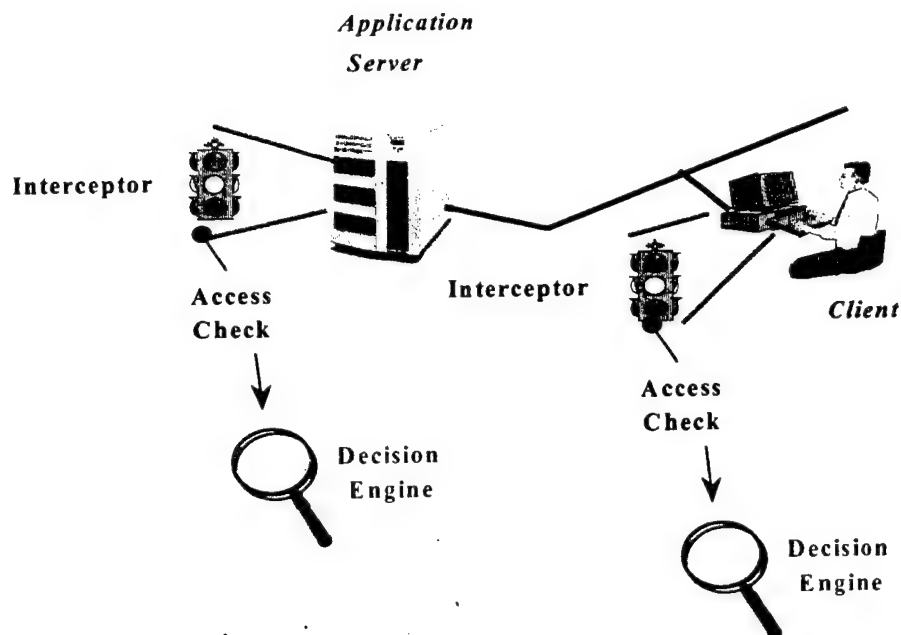


Figure 8. CORBA Access Control Using Interceptors

As shown in Figure 8, the approach used the interceptor feature supported by both Iona Technology's Orbix product and Visigenics' Visibroker. Interceptors were implemented on both the client and server sides that would intercept method invocations. The interceptors would query the decision server and either allow or disallow the invocation based on the policy specified in a decision server.

On the client side the user's credentials were inserted, and on the server side, these credentials were checked to determine whether access would be allowed. Three different scenarios were prototyped:

- Visibroker Interceptors using SSL to provide the credentials and the OGRI's Pledge as the decision engine
- Visibroker Interceptors using IIOP Service Context to pass the credentials and Pledge or Secure Computing's Lock6 as the decision engine
- Orbix Filters using their "Interface" Context to pass the credentials and Lock6 as the decision engine.

As part of IFD 1.2, Secure Computing also integrated the Napoleon policy management tool with the Interceptor/Pledge system to provide a means for defining the policy that the Pledge decision engine would support and the interceptor would enforce. The demonstration showed how the integrated system could be used to control access to the Plan Server at that time being developed as one of DARPA's

composable services.

2.4.2.1 Future Directions

The CORBA access control work was aimed at providing additional security that CORBA vendors were not currently providing. Since the work was initiated, there have been several commercial vendors who have developed products that implement some or all of the security functionality that is present in CORBASEC level 2 and which our work addressed. Any future work in this area should build on this commercially available technology.

2.4.2.2 Lessons Learned

An issue that arose that goes beyond CORBA access control is what the best architectural approach is for a policy decision server: centralized or distributed. With a centralized approach, every access check is directed to the decision server. The advantage of this approach is that it is more easily implemented and may be easier to administer. The disadvantages are performance and reliability, since it provides a single point of failure. With the distributed approach, the decision server is decomposed into components that are either more location specific or application specific. Such an approach is harder to implement, but should provide better performance and reliability. In our opinion some version of the distributed approach is preferred. In either case, for performance reasons caches should be used at the enforcement mechanisms, if possible, to avoid having to make out-calls to the decision server on each access control check.

2.4.3 DCOM Study

DCOM occupies a central part in Microsoft's vision for distributed object oriented computing. Microsoft has positioned DCOM as the "glue" which binds each machine in the network together to form a cooperating whole, with goals of increased performance, robustness, and availability while reducing costs. While such openness is laudable, how does DCOM perform in an environment where an adversary is actively seeking to disrupt the network harmony? Has Microsoft provided sufficient strength in its security mechanisms supporting DCOM to thwart such an adversary? The DCOM study[11] undertaken under this program was aimed at answering these questions and suggesting possible enhancements to DCOM security based on IA technologies.

During the course of the study we examined DCOM from a security perspective, both architecturally and at the implementation level, but we did not attempt a detailed vulnerability analysis. Rather, the focus was to reveal what security features Microsoft has built into its product. DCOM is highly ubiquitous; it is bundled and distributed with every Windows NT system (including 2000), and available at no cost for Windows 9x systems. While Microsoft's DCOM implementation does provide some strong security features, including encryption, authenticated communications and integrity mechanisms to prevent tampering, the base of these features is anchored in the underlying Windows security architecture, such as the NT Lan Manager.

Windows 2000 introduced stronger mechanisms such as Kerberos V5. Such mechanisms are indeed sufficient for many environments.

But what about our determined, well-heeled adversary? Rather than trying to point out flaws in the Lan Manager and Kerberos protocols (which have been the focus of other studies), we postulate that our adversary has succeeded. In such an eventuality, how can we respond in a DCOM environment where the security functionality is buried in the Microsoft source code? This study therefore focused its efforts on highlighting areas where existing gaps and gaps exposed by an adversary can be quickly filled by hooking alternate or additional security mechanisms into the Microsoft architecture, without waiting for some future product release to close the vulnerability.

DCOM relies heavily on the implementation of the underlying Security Service Provider (SSP) to securely access DCOM objects. It is absolutely critical that the protocols used in the SSP be sufficient for the environment where DCOM will be employed, or all other attempts to bolster security will be in vain. The SSP is the foundation on which all other security functions rest. Microsoft does publish the specification for writing an SSP, but it is a substantial undertaking. One could author a new SSP, such as one adding support for Type I cryptography, but the effort must be well planned and executed in advance of the expected need or this approach will fail due to its untimely deployment.

Other less drastic approaches documented in the study rely on and assume the integrity of the SSP.

We examined a variety of techniques to incorporate new security functionality into DCOM, and examined areas where existing infrastructure could be used to limit risk. From this examination we concluded the following:

- The garbage collection mechanism of the DCOM protocol is vulnerable without employing the authentication and integrity features; both administrator and component developer must be cognizant of the relative risk versus the tradeoff of the performance penalty incurred when the security features are enabled.
- A role based policy can be constructed using the Microsoft Transaction Server (MTS), but can only be used for server side controls. A finer grained security policy can be constructed using Decision Engines from the IA program (Lock6, Pledge and OODTE) in conjunction with the Napoleon tool for policy construction and management.
- Managing a DCOM system is shown to be a frightful undertaking. The Microsoft tools do not provide sufficient insight into the workings of the DCOM configuration to get a good understanding of the security policy currently being enforced. We show how the Napoleon tool can be used to aid in understanding of an MTS role based policy and how it can be used to configure the IA decision engines.
- Interceptors can be used to add additional security functionality using a security component approach called a hook. The component developer is freed to

concentrate on the functionality of the component and relies on the security features transparently enforced on its behalf by the interceptor.

- Firewalls can be used to support the DCOM protocol. We also provided suggestions for additional functionality which would increase the security of the system.
- NT shared library wrappers can be used to control which local shared library entry points can be accessed by a DCOM component.

2.4.3.1 Future Directions

Although the DCOM study suggested possible areas where IA technology might be used to increase DCOM security, no further work in actually developing these areas was undertaken for a number of reasons. The primary one was that program resources were redirected to other work that was of more interest and had more potential payoff. However, another major reason for not pursuing additional work in this area is that others on the program were already doing some of it (for example, the NT library wrappers). Also there was some doubt as to the actual value that the work might eventually have. For example, while a DCOM firewall proxy could be implemented, as in the case with the IIOP proxy, there are very few Internet applications that use DCOM as their protocol. Hence, there is little demand for a DCOM firewall proxy. Similarly, while additional access control could be done using some form of interceptor, there is a trade-off between the additional security this would provide as opposed to the additional policy management complications it would present. Windows NT and 2000 already have fairly complex security features, which are often not managed correctly, and adding more may only add to the confusion.

2.4.3.2 Lessons Learned

The primary lesson learned from the study is that the Microsoft DCOM and security environment is relatively complex and inserting additional mechanisms into the system that will reliably add additional security will not be easy. There are a variety of ways to access resources in this environment and identifying and controlling them all in a consistent manner will require different, coordinated mechanisms.

In addition, as mentioned in the above Future Directions section, many of the lessons on the IIOP proxy also apply to a DCOM proxy as do the lessons learned for CORBA access control regarding decision servers.

3 Summary

Secure Computing's IAT program successfully contributed to a range of technology areas that were of interest to DARPA. These areas included security architecture, role based access control, single sign-on mechanisms, and distributed object security. Much of the work from the program continues to progress, either as features in commercial products or as the basis for on-going work in other programs. While some of the work has been overtaken by advances in the fast-paced commercial world, even here, the insights gained on the program are valuable in identifying future research areas. Overall, the program achieved significant progress in a variety of security areas.

4 References

- [1] Secure Computing Corporation. Checkmate Architecture Model, Information Assurance Technologies for the GCCS LES Program, CDRL A006. Sept 2000.
- [2] Secure Computing Corporation. Checkmate User Guide, Information Assurance Technologies for the GCCS LES Program, CDRL A011. Sept 2000.
- [3] Tom Markham, Tomo Foote-Lennox, David Apostol, Alan Dowd, Raymond Lu, and Richard O'Brien. Checkmate Network Security Modeling. *Proceedings of Milcom 2000*, Oct 2000.
- [4] Secure Computing Corporation. RBAC Framework Design, Information Assurance Technologies for the GCCS LES Program, CDRL A012. Sept 2000.
- [5] Secure Computing Corporation. Napoleon User Guide, Information Assurance Technologies for the GCCS LES Program, CDRL A013. Oct 2000.
- [6] Dan Thomsen, Dick O'Brien, and Jessica Bogle. Role based access control framework for network enterprises. *Proceedings of the Fourteenth Annual Computer Security Applications Conference*, 1998.
- [7] D. Thomsen, R. C. O'Brien. Network Application POLicy EnvirONment (NAPOLEON). *Proceedings of the fourth ACM RBAC conference*, pp. 134-142, Oct 28-29 1999.
- [8] C. Payne, D. Thomsen, J. Bogle, R. C. O'Brien. NAPOLEON: A recipe for workflow. *Proceedings of the Fifteenth Annual Computer Security Applications Conference*, pp. 145-152, Dec 1999.
- [9] D. Thomsen and R. O'Brien. Layered Security Policy Management Using Napoleon. . *Proceedings of Milcom 2000*, Oct 2000.
- [10] Secure Computing Corporation. Single Sign-on Plugin User Guide, Information Assurance Technologies for the GCCS LES Program, CDRL A009. Sept 2000.
- [11] Secure Computing Corporation. DCOM Security Report, Information Assurance Technologies for the GCCS LES Program, CDRL A010. Sept 2000.
- [12] Secure Computing Corporation. CORBA IIOP Proxy Design Report, Information Assurance Technologies for the GCCS LES Program, CDRL A007. July 2000.
- [13] Secure Computing Corporation. CORBA IIOP Proxy User Guide, Information Assurance Technologies for the GCCS LES Program, CDRL A008. July 2000.
- [14] Secure Computing Corporation. Policy Workshop I Report.
- [15] Tom Markham, Dwight Colby, Mary Denz. Security Modeling in the COTS Environment. *New Security Paradigms Workshop*, 1999.

- [16] CERT Coordination Center <http://www.cert.org>
- [17] Bugtraq <http://www.geek-girl.com/resources.html>
- [18] RootShell <http://www.rootshell.com/beta/news.html>
- [19] D. Sterne, G. Tally, C. McDonell, P. Pasturel, D. Sames, D. Sherman, E. Sebes, "Scalable Access Control for Distributed Object Systems," *Proceedings of the 8th USENIX Security Symposium*, Washington, DC, August 1999.
- [20] T. Fraser, L. Badger, M. Feldman "Hardening COTS Software with Generic Software Wrappers" *Proceedings of the 1999 Symposium on Security and Privacy*, pp. 2-16, May 1999.
- [21] M. Zurko, R. Simmon, "User-Centered Security," *New Security Paradigms Workshop*, September 1996.

***MISSION
OF
AFRL/INFORMATION DIRECTORATE (IF)***

The advancement and application of Information Systems Science and Technology to meet Air Force unique requirements for Information Dominance and its transition to aerospace systems to meet Air Force needs.